VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Nástroj pro analýzu časových řad pomocí sítí

Tool for Time Series Analysis by Means of Networks

2018 Pavol Urban

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Pavol Urban**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Nástroj pro analýzu časových řad pomocí sítí
Tool for Time Series Analysis by Means of Networks

Jazyk vypracování: čeština

Zásady pro vypracování:

Sítě jsou všude okolo nás. Potkáváme je na každém rohu. Při cestě do školy či zaměstnání (silniční sítě, železniční sítě), při styku s ostatními lidmi (sítě přátel, sítě lidských kontaktů), při komunikaci (komunikační sítě, emailové sítě) a mnoho dalších. Pro analýzu takovýchto sítí existuje mnoho nástrojů a knihoven. Tyto nástroje a knihovny také můžeme využít pro analýzu dat, které na první pohled neskrývají žádnou síť. Jedním z příkladů můžou být třeba časové řady.

Úkolem této práce je nastudovat problematiku sítí (komplexních sítí) a naprogramovat nástroj, který umožní převod časové řady na síť a její následnou analýzu.

Cíle této práce je možné shrnout v těchto bodech:
1. Nastudovat a popsat problematiku sítí (komplexních sítí).
2. Nastudovat a popsat možnosti převodu časových řad na síť.
3. Naprogramovat a popsat nástroj, který umožní: načtení časové řady, převod na síť a základní analýzu vzniklé sítě.
4. Nástroj vhodně otestovat.
5. Porovnat 5 časových řad dodaných vedoucím této práce.

Seznam doporučené odborné literatury:

[1] BOCCALETTI, Stefano, et al. Complex networks: Structure and dynamics. Physics reports, 2006, 424.4: 175-308.
[2] LACASA, Lucas, et al. From time series to complex networks: The visibility graph. Proceedings of the National Academy of Sciences, 2008, 105.13: 4972-4975.
[3] NEWMAN, Mark. Networks: an introduction. 2010. United Slates: Oxford University Press Inc., New York
[4] SMALL, Michael; ZHANG, Jie; XU, Xiaoke. Transforming time series into complex networks. In: International Conference on Complex Sciences. Springer Berlin Heidelberg, 2009. p. 2078-2089.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Lukáš Tomaszek**

Datum zadání: 01.09.2017

Datum odevzdání: 30.04.2018

doc. Ing. Jan Platoš, Ph.D.
vedoucí katedry

prof. Ing. Pavel Brandštetter, CSc.
děkan fakulty

Prehlasujem, že som túto bakalársku prácu vypracoval samostatne. Uviedol som všetky literárne pramene a publikácie, z ktorých som čerpal.

V Ostrave 30. apríla 2018 .....

Rád by som na tomto mieste vyjadril poďakovanie vedúcemu tejto práce Ing. Lukášovi Tomaszkovi za jeho ústretový prístup a odbornú pomoc, ktoré umožnili jej vznik.

Abstrakt

Táto práca je zameraná na analýzu časových rád za pomoci sietí. Sú v nej popísané základné termíny z teórie grafov, ďalej metrika sietí, časové rady a spôsoby, ktorými je takéto rady možné transformovať na siete. Tieto teoretické základy sú následne využité pri implementácii nástroja, ktorý umožňuje analýzu časových rád takým spôsobom, že ich transformuje na sieť a následne vypočíta vybrané vlastnosti týchto novovzniknutých sietí.

Kľúčové slová: časová rada, graf, sieť, metrika sietí

Abstract

This bachelor thesis is focused on analysis of time series by means of networks. There are described basic terms from graph theory. There are also described network metrics, time series and ways to tranform these time series into the networks. These theoretical basics are subsequently used in implementation of tool, which is able to analyze the time series by converting it into the networks and to count chosen properties of these newly-created networks.

Key Words: time series, graph, network, network metrics

Obsah

Zoznam použitých skratiek a symbolov

HVA	–	Horizontal visibility algorithm
HVG	–	Horizontal visibility graph
JUNG	–	Java Universal Network/Graph Framework
NVA	–	Natural visibility algorithm
NVG	–	Natural visibility graph

Zoznam obrázkov

Zoznam tabuliek

Zoznam výpisov zdrojového kódu

1 Úvod

Existuje mnoho systémov zostavených z jednotlivých komponentov, ktoré sú nejakým spôsobom prepojené. Príkladom môže byť Internet, ktorý je kolekciou zariadení prepojených dátovými spojeniami alebo ľudské spoločenstvá, čo sú v skutočnosti len kolekcie ľudí prepojených rôznymi druhmi vzťahov.

Na tieto systémy môžeme z hľadiska analýzy nahliadať rôzne. Niektorí ľudia študujú vlastnosti jednotlivých komponentov (ako pracuje počítač, ako sa správa, či cíti človek). Iní zase svoju pozornosť zamerajú na analýzu podstaty prepojení (komunikačné protokoly Internetu alebo dynamika ľudských priateľstiev). No existuje aj tretí aspekt prístupu k analýze takýchto systémov, ktorý je niekedy prehliadaný, ale za to takmer vždy rozhodujúci o správaní systému a to je vzor spojení medzi komponentmi. Tento vzor v danom systéme môže byť reprezentovaný sieťou.

Bez ohľadu na to, či si to uvedomujeme alebo nie, sme obklopení sieťami. Ak niekam cestujeme, môžeme využiť železničnú či cesntnú sieť [18]. Ak sa dostávame do kontaktu s ľuďmi, využívame siete rôznych spoločenských vzťahov [13, 19]. Osobitnou kategóriou v oblasti medziľudských kontaktov, sú sociálne siete [5, 6, 7, 11, 22], kde je už priamo z názvu zrejmé, že ide opäť o sieť.

Pre pochopenie vzťahov v týchto sieťach, je potrebné tieto siete analyzovať. Na analýzu sietí bolo vyvinutých množstvo nástrojov a knižníc, ako sú napríklad [1, 2, 4, 12, 14] a mnoho ďalších. Tieto nástroje a knižnice nachádzajú uplatnenie aj pri analýzach dát, ktoré na prvý pohľad žiadnu sieť neobsahujú. Zaujímavým príkladom dát, v ktorých sieť nie je evidetná, sú časové rady [26, 29]. A práve o ich analýzu pomocou sietí sa budeme zaujímať v tejto práci.

V prvej kapitole sa budeme venovať teórii grafov, ktorá je základným kameňom tejto práce. Ako si v nej povieme, sieť nie je nič iné ako graf. Preto ak chceme analyzovať siete, musíme byť oboznámení s problematikou teórie grafov. V druhej kapitole sa dozvieme, ako je možné takúto sieť, graf reprezentovať, čo je dôležité pre samotnú implementáciu. V tretej kapitole sa budeme zaoberať metrikou sietí, teda vlastnosťami sietí. Tieto vlastnosti nám prezrádzajú informácie o sieťach a podľa nich môžeme následne siete porovnávať, hľadať v nich podobnosti či odlišnosti. V nasledujúcej kapitole si zadefinujeme časovú radu a ukážeme si metódy, ktoré ju dokážu transformovať na sieť. V ďalšej kapitole si povieme o nástroji, ktorý všetky poznatky z predošlých kapitol spojí v program, ktorý umožní analýzu časových rád pomocou sietí. A samozrejme, tiež si ukážeme výsledky, ktoré tento program umožňuje získať. Motivácia pre túto bakalársku prácu je práve získanie týchto výsledkov.

2 Teória grafov

Sieť, v jej najzákladnejšej forme, je množina bodov navzájom poprepájaných čiarami. V matematickej literatúre je sieť často označovaná termínom **graf**. Preto teoretický základ k štúdiu sietí predstavujú práve grafy, kde body sú nazývané **vrcholy** (niekedy tiež uzly) a čiary sú **hrany**.

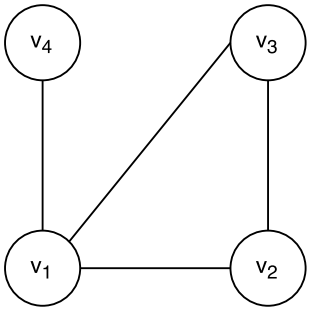
V tejto práci sa budeme držať klasickej konvencie, aká je používaná v matematickej literatúre a síce, že počet uzlov v sieti budeme značiť písmenom n a počet hrán písmenom m.

2.1 Graf

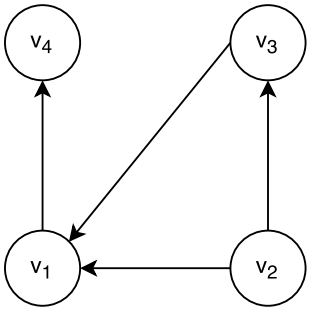
Graf G je usporiadanou dvojicou G = (V,E), kde V predstavuje neprázdnu množinu vrcholov a E je množina hrán. Je to množina (niektorých) dvojprvkových podmnožín množiny V.

Z tejto definície vyplýva, že väzba medzi 2 vrcholmi je obojstranná - jedna hrana jednoducho prepája dva uzly a keďže nemá smer, nedokážeme určiť, z ktorého uzlu vychádza a do ktorého vedie. Preto v tomto prípade hovoríme o **neorientovanom grafe** (pozri obrázok 1).

Vychádzajúc z definície [16], pod pojmom **orientovaný graf** (tentokrát ho vidíme na obrázku 2) rozumieme usporiadanú dvojicu G = (V,E), kde V je množina vrcholov a množina orientovaných hrán je E $\subseteq$ V $\times$ V. Orientovaný a neorientovaný graf, za predpokladu, že sú vizualizované, jednoducho rozlíšime tak, že v orientovanom grafe sú hrany znázornené šípkami, zatiaľ čo v neorientovanom sú uzly spojené iba čiarami (ako je to zrejmé pri pohľade na obrázky 1 a 2). V našom nástroji pre analýzu časových rád pomocou sietí sa ale s orientovanými grafmi nestretneme, pretože, ako si povieme v kapitole o prevodoch časových rád, všetky tieto prevody vytvárajú neorientované grafy. Nestretneme sa ani s orientovaným grafom, ktorý prideľuje hranám hodnoty, a tak bližšie určuje ich významnosť.



Obr. 1: Neorientovaný graf



Obr. 2: Orientovaný graf

2.1.1 Podgraf

Definícia [16] hovorí, že graf $G_1(V_1,E_1)$ je podgrafom pôvodného grafu G(V,E), ak $V_1 \subseteq V \wedge E_1 \subseteq E$. Zapíšeme $G_1 \subseteq G$. Teda môžeme povedať, že podgraf vzniká odstraňovaním hrán a vrcholov z pôvodného grafu. Nesmie však nastať situácia, že by nejaká hrana bola pripojená len k jednému, prípadne k žiadnemu uzlu.

3 Reprezentácia sietí, grafov

V predchádzajúcej kapitole sme si priblížili problematiku grafov. Na to, aby sme s nimi mohli pracovať vo výpočtovej technike, ich však musíme vedieť vhodne reprezentovať. Pri rozhodovaní o tom, ako budeme graf reprezentovať musíme zvážiť pre a proti tak, aby bol program, čo najefektívnejší. Niektoré techniky môžu byť veľmi jednoduché z hľadiska implementačnej náročnosti, no o to vyššie budú ich nároky na pamäť.

3.1 Incidenčná matica

Existuje viacero spôsobov ako matematicky reprezentovať sieť. Jedným z najvyužívanejších je incidenčná matica, ktorú si označíme A. Incidenčná matica má rozmery $n \times m$, teda riadky tvoria uzly a stĺpce hrany. Pokiaľ tvoríme incidenčnú maticu pre neorientovaný graf, nájdeme v nej hodnoty 0 a 1 (tak ako to môžeme vidieť v matici A). Hodnota 1 sa v matici nachádza na mieste, kde je príslušný uzol s príslušnou hranou spojený. V prípade, že je na nejakej pozícii 0, znamená to, že daný uzol s danou hranou spojený nie je.

Matica A je vytvorená pre graf z obrázku 1 a platí pre ňu nasledovná organizácia:

- Riadky v maticiach tvoria uzly v poradí zhora: v_1, v_2,v_3, v_4.
- Stĺpce v matici tvoria hrany v poradí zľava: v_1-v_2, v_1-v_3, v_1-v_4, v_2-v_3 .

$$A = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

3.2 Matica susednosti

Ďalším spôsobom ako reprezentovať sieť je matica susednosti, ktorú si označíme B. Tá je rozmerov $n \times n$. V ukážkovej matici B, budeme značiť podľa indexu uzlu, teda prvým uzlom je v_1 druhým v_2 atď. V matici susednosti vytvorených pre neorientovaný graf, sa môžeme stretnúť s hodnotami 1 a 0. Hodnotu 1 nájdeme na pozíciach, kde sú príslušné uzly spojené. Ak medzi uzlami neexistuje hrana, nájdeme v matici 0.

Ak nás zaujíma či v matici existujú slučky, kde je uzol pripojený sám na seba, venujeme pozornosť hlavnej diagonále, teda priamke smerujúcej z ľavého horného rohu do pravého dolného rohu. Ak sú na nej samé nuly, potom v grafe nie je žiadny uzol spojený sám so sebou.

$$B = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

3.3 Zoznam susedov

Problémom reprezentovania grafov cez incidenčné matice, prípadne matice susednosti, je veľká pamäťová náročnosť pri väčšom počte uzlov.

Výhodnejším spôsobom je vytvoriť zoznam susedov, kde pre každý uzol udržiavame v tabuľke jeho susedov tak, ako to môžeme vidieť v tabuľke 1, ktorá bola vytvorená pre graf z obrázku 1.

Tabuľka 1: Zoznam susedov pre jednotlivé uzly z obrázku 1

Vrchol	Zoznam susedov
v_1	v_2, v_3, v_4
v_2	v_1, v_3
v_3	v_1, v_2
v_4	v_1

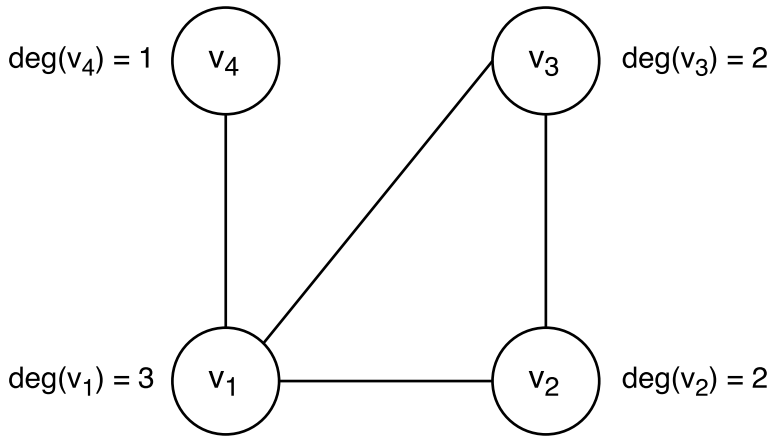
4 Metrika sietí

Zo sietí je možné vyčítať rôzne informácie, vďaka ktorým môžeme zachytiť ich osobité vlastnosti. V tejto kapitole si preto povieme o niektorých z vlastností sietí, ktoré budú pre túto bakalársku prácu relevantné.

4.1 Stupeň (degree)

Stupeň uzla v je tak, ako je uvedné v [23], súčtom k nemu pripojených hrán. Značíme *deg(v)*. Napríklad na obrázku 3, je uzol v_1 pripojený ku všetkým ostatným uzlom (v_2,v_3 aj v_4) v grafe, teda má stupeň 3. Podobne by sme mohli sledovať pripojené uzly aj pri zvyšných uzloch (tak, ako to môžeme vidieť na na obrázku 3.

Pri orientovanom grafe ešte rozlišujeme vstupný $deg_{in}(v)$ a výstupný $deg_{out}(v)$ stupeň uzla. $Deg_{in}(v)$ nám hovorí o tom, koľko hrán do uzla vchádza (v grafe smeruje šípka smerom k uzlu) a $deg_{out}(v)$ zase o tom koľko hrán z neho vychádza (v grafe smeruje šípka smerom od uzla).



Obr. 3: Ukážka výpočtu stupňa uzlov pre neorientovaný graf

4.2 Distribúcia stupňov

Ide o jednu z najzákladnejších vlastností sietí. V podstate ide o frekvenciu výskytu uzlov daného stupňa. Pre daný stupeň uzla k ju vypočítame rovnicou $p_k = \frac{i}{n}$, kde p_k je početnosť výskytu uzlov stupňa k, n je ako obvykle celkový počet uzlov v sieti a i je počet uzlov stupňa k.

4.3 Hustota

Hustota siete udáva pomer existujúcich hrán medzi uzlami voči všetkým možným potenciálnym spojeniam medzi uzlami [23].

Na to, aby sme mohli hustotu vypočítať, potrebujeme najskôr zistiť počet všetkých potenciálnych spojení P_c. Tento počet vypočítame vzťahom z rovnice (1).

$$P_c = \frac{n * (n-1)}{2} \tag{1}$$

Keď poznáme hodnotu P_c, potom jednoducho môžeme vypočítať aj hustotu ρ podľa rovnice (2).

$$\rho = \frac{\mathrm{m}}{P_c} \tag{2}$$

Hodnota hustoty sa nachádza v intervale $0 \leq \rho \leq 1$. Čo je zrejmé, pretože nemôže existovať menej než 0 hrán a zároveň počet hrán, ktoré sa nachádzajú v sieti, nikdy nebude vyšší ako P_c.

4.4 Sled, vzdialenosť, ťah, cesta, najkratšia cesta

Ústredný koncept pri teórii sietí (grafov) je, že sledujeme dostupnosť dvoch rozdielnych uzlov v danej sieti. Musíme si ale uvedomiť, že dva uzly, ktoré nie sú priamo prepojené hranou, môžu byť napriek tomu navzájom dosiahnuteľné. A to prostredníctvom ďalších uzlov. Preto si teraz nadefinujeme podľa [3] pojmy: sled, vzdialenosť, ťah, cesta a najkratšia cesta.

Sled medzi uzlami v_1 a v_2 je sekvencia uzlov a hrán (každý uzol obsiahnutý v sekvencii musí byť pripojený na predošlý), ktorá začína v uzle v_1 a končí v uzle v_2.

Ťah je taký sled, kde sa v sekvencii neopakujú hrany, zatiaľ čo **cesta** je sledom, kde sa neopakujú uzly.

Vzdialenosť uzlov je definovaná ako počet hrán v slede. Vzdialenosť budeme označovať ako d_{uv}. Cesta s minimálnou vzdialenosťou medzi dvoma uzlami je nazývaná **najkratšia cesta**, v odbornej literatúre je používaný taktiež termín characteristic path. Ak medzi dvoma uzlami neexistuje žiadna cesta, vzdialenosť položíme rovnú ∞ (uzol je z daného uzlu nedosiahnuteľný).

4.5 Lokálny a globálny zhlukovací koeficient

Pod pojom zhlukovací koeficient rozumieme pravdepodobnosť, že dvaja susedia uzla sú si susedmi aj navzájom. V skutočnosti nám zhlukovací koeficient meria hustotu trojuholníkov (cyklus na troch uzloch) vyskytujúcich sa v sieti.

Ako sa dočítame v [23] môžeme definovať zhlukovací koeficient pre jeden uzol. Pre uzol v môžeme tento koeficient vypočítať podľa rovnice (3).

$$C_v = \frac{\text{počet párov susedov uzla v, ktoré sú spojené}}{\text{počet párov susedov uzla v}} \tag{3}$$

To znamená, že na výpočet C_v musíme prejsť cez všetky rozdielne páry uzlov, ktoré sú susedmi uzla v v danej sieti, spočítať počet takýchto párov, ktoré sú navzájom prepojené a to celé podeliť celkovým počtom párov. C_v niekedy nazývame aj **lokálny zhlukovací koeficient**.

Druhým druhom zhlukovacieho koeficientu, ako ho navrhli Watts a Strogatz [27] je tzv. **globálny zhlukovací koeficient**. Ten je určený na výpočet zhlukovacieho koeficientu pre celú sieť, ako priemer lokálnych zhlukovacích koeficientov jednotlivých uzlov, čo môžeme vidieť v rovnici (4).

$$C_{ws} = \frac{1}{n} \sum_{i=1}^{n} C_v \tag{4}$$

4.6 Priemerná dĺžka najkratšej cesty

Priemerná dĺžka najkratšej cesty je daná priemerom všetkých najkratších ciest medzi všetkými vrcholmi. Matematicky to môžeme zapísať rovnicou (5).

$$d_{avg} = \frac{\sum_{u}^{n} \sum_{v:v \neq u}^{n} d_{uv}}{n} \tag{5}$$

4.7 Priemer

Priemer siete je najdlhšia najkratšia vzdialenosť medzi ľubovoľnými dvoma uzlami v rámci jednej komponenty siete.

4.8 Centrality

Centralita je mimoriadne dôležitým pojmom pri zisťovaní špecifík siete. Hľadá odpoveď na otázku: Ktoré uzly sú v danej sieti najdôležitejšie? V tejto práci si predstavíme len niektoré centrality, podľa toho ako sú definované v [23].

4.8.1 Degree centrality

Zrejme najjednoduchšou centralitou je tzv. degree centrality. Degree (teda stupeň) uzla určuje koľko hrán je pripojených k danému uzlu. Napriek tomu o akú jednoduchú vlastnosť ide, môže byť veľmi užitočná. Napr. ak si ako uzly predstavíme ľudí a vzťahy medzi nimi budeme reprezentovať hranami, je evidentné, že uzol s najvyšším stupňom (v našom prípade človek s najviac kontaktmi) je dôležitý. Samozrejme, problémom je, že nevieme nič o tom, ako významné sú tieto kontakty samé o sebe (teda napr. či aj oni majú pripojenie na ďalších ľudí).

Degree centralitu počítame podľa rovnice (6).

$$C_v^{deg} = \frac{deg(v)}{n-1} \tag{6}$$

Pričom *deg(v)* je stupeň vrcholu, pre ktorý chceme vypočítať degree centralitu a n, ako sme si už zaviedli podľa konvencie, je počet vrcholov v sieti.

4.8.2 Closeness centrality

Closeness centrality meria priemernú vzdialenosť z uzla do ostatných uzlov. Vypočítame ju rovnicou (7).

$$C_v^{clo} = \frac{1}{n-1} \sum_{v \neq u} d_{uv} \tag{7}$$

d_{uv} je vzdialenosť medzi uzlami u a v a je daná ako priemer najkratších ciest z daného uzlu do všetkých ostatných uzlov.

4.8.3 Betweenness centrality

Ďalšou z centralít je betweenness centrality. Vychádza z toho, koľko najkratších ciest prechádza cez daný uzol. Samozrejme, čím viac najkratších ciest prechádza cez daný uzol, tým je tento uzol v príslušnej sieti dôležitejší. Betweenness centralitu vypočítame rovnicou (8).

$$C_v^{bet} = \sum_{u \neq v, w \neq v} \frac{p_{uw}(v)}{p_{uw}} \tag{8}$$

V rovnici (8) p_{uw} znamená počet ciest z uzlu u do uzla w a $p_{uw}(v)$ je počet ciest prechádzajúcich uzlom.

4.8.4 Eigenvector centrality

Eigenvector je rozšírením jednoduchej degree centrality. Degree centralita ocení uzol pričítaním jedného centrality bodu za každého suseda. Problémom tejto metódy je, že hoci nie všetci susedia sú si rovní, teda niektorý uzol je významný (má mnoho ďalších susedov) a iný nie je (ma málo susedov, prípadne nemá žiadnych ďalších), za všetkých je rovnaké ocenenie.

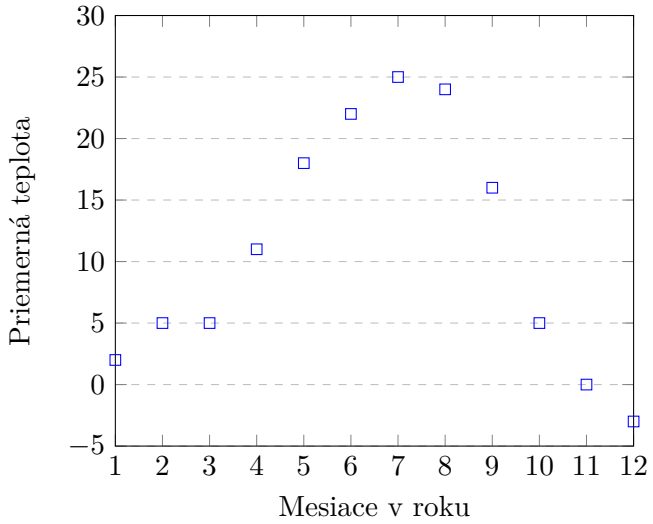
Tento problém rieši eigenvector centrality. Namiesto oceňovania uzlov tak, že uzol dostane jeden bod za každého suseda, eigenvector centralita dáva každému uzlu skóre úmerné súčtu skóre jeho susedov, teda pridelená hodnota rastie s rastúcou významnosťou uzla.

5 Časové rady a ich prevody na sieť

Podľa [8] definujeme časovú radu ako množinu pozorovaní x_t, kde každé jedno pozorovanie je zaznamenané v špecifikovanom čase t.

Diskrétna časová rada je taká časová rada, kde množina časov T_0, v rámci ktorej sú robené pozorovania, je diskrétnou množinou, ako je tomu napríklad v prípade, keď sú pozorovania robené vo fixných časových intervaloch.

S časovými radami sa môžeme stretnúť v rôznych oblastiach života ako sú napríklad veda, sociológia či ekonómia. Pre lepšiu predstavu - príkladom časovej rady môže byť napríklad vývoj priemernej teploty v jednotlivých mesiacoch počas roka ako to môžeme názorne vidieť na obrázku 4. Údaje sú len ilustratívne a nereflektujú žiadne reálne merania. V tejto bakalarskej práci, budeme spracovávať časové rady, ktoré obsahujú oveľa väčšie množstvo hodnôt, než naša ilustratívna, ktorú vidíme na obrázku 4. Konkrétne pôjde o 4 000 hodnôt v rámci jedinej časovej rady.



Obr. 4: Vývoj priemernej mesačnej teploty počas jedného kalendárneho roka (údaje sú len ilustratívne)

5.1 Visibility graph

V tejto podkapitole si predstavíme metódu, ktorá je využívaná pri analýze časových rád - visibility graph (graf viditeľnosti). Tento algoritmus prevádza časovú radu na sieť. Hlavnou myšlienkou je zistiť, do akej miery nám môže byť teória grafov užitočná pri charakterizovaní časovej rady. Takto vzniknutá sieť dedí niektoré vlastnosti príslušnej časovej rady a jej ďalšie štúdium nám odhaľuje netriviálne informácie o pôvodnej časovej rade.

5.1.1 NVG

Skratka NVG pochádza z anglického Natural visibility graph (prirodzený graf viditeľnosti) a odteraz ju budeme používať. Rovnako budeme používať aj skratku NVA, čo je skratka pre Natural visibility algorithm.

Nech $\{x(t_i)\}_{i=1...N}$ je časová rada N dát. V princípe podľa [17], NVA priraďuje každej jednej hodnote z danej časovej rady uzol v NVG. Teda ak časová rada obsahuje 20 hodnôt, vo výslednom grafe bude taktiež 20 uzlov. Ak chceme zistiť či sú dva uzly u a v spojené, musíme pracovať s ich hodnotami $x(t_u)$ a $x(t_v)$. Predstavme si, že 20 hodnôt v časovej rade tvorí stĺpcový diagram (čím vyššia hodnota tým vyšší stĺpec). Dva uzly sú v grafe spojené ak môžeme vykresliť medzi ich stĺpcami priamu čiaru tak, že nepretneme žiadny stĺpec s hodnotou $x(t_k)$ v časovej rade, ktorý leží medzi nimi.

Formálne, ako je uvedené v [17], u a v budú v grafe spojené ak je splnené nasledujúce geometrické kritérium:

$$x(t_k) < x(t_u) + (x(t_v) - x(t_u))\frac{t_k - t_u}{t_v - t_k}$$

Z toho môžeme ľahko overiť, že graf získaný z príslušnej časovej rady bude vždy:

1. Súvislý: každý uzol vidí aspoň jeho najbližšieho suseda (vľavo a vpravo).
2. Neorientovaný: spôsob akým je algoritmus navrhnutý nedefinuje smer spojenia medzi uzlami.
3. Nemenný pri zmene veľkostí oboch osí (horizontálnej i vertikálnej) .

5.1.2 HVG

Horizontal visibility graph (v preklade Horizontálny graf viditeľnosti) budeme nazývať ďalej len HVG. S HVG súvisí takzvaný Horizontal visibility algorithm, ktorý slúži na vytvorenie HVG. Tento algoritmus budeme označovať jeho anglickou skratkou, ktorá sa zároveň najčastejšie vyskytuje aj v odbornej literatúre, a teda HVA.

V [24] môžeme nájsť nasledujúce alternatívne kritérium na zostrojenie grafu viditeľnosti: nech $\{x_i\}_{i=1...N}$ je časová rada N dát. HVA potom priraďuje každej hodnote tejto rady uzol v HVG. Dva uzly u a v sú v grafe spojené ak môžeme nakresliť horizontálnu čiaru v časovej rade, ktorá spája x_u a x_v tak, že nedôjde k pretnutiu žiadnych dát medzi nimi.

Teda u a v budú spojené ak v časovej rade platí toto geometrické kritérium:

$$x_u, x_v > x_z$$

pre všetky z také, že $u < z < v$. Tento algoritmus je zjednodušením NVA. V skutočnosti HVG je vždy podgrafom NVG pre danú časovú radu. A rovnako spĺňa body vymenované pri NVG (teda bude tiež súvislý, neorientovaný a nemenný pri zmenách veľkostí oboch osí).

5.2 Proximity networks

Visibility graph nie je jedinou metódou prevodu časovej rady na sieť. Ďalšia skupina konceptov ako robiť takéto prevody je nazývaná **proximity networks**. Tento názov budeme používať v jeho anglickom originále. Prevody môžu byť robené rozličnými spôsobmi a teda sú aj rôzne typy takýchto proximity networks napríklad: cycle network, reccurence network, correlation network (opäť ponecháme anglické termíny). Všetky tieto metódy sú ale charakterizované 2 všeobecnými vlastnosťami:

1. Výsledné siete sú nemenné pri zámene ich uzlov v matici susedností.

2. Proximity networks sú priestorové siete.

5.2.1 Correlation networks

V [28] sa dočítame o spôsobe prevodu časovej rady na sieť. Začneme tým, že si jednotlivé hodnoty časovej rady označíme ako $\{S_1, S_2, ..., S_n\}$ Spojením začiatku a konca tejto časovej rady obdržíme všetky možné segmenty o dĺžke L. Môžeme zapísať:

$$\{T_m = S_m, S_{m+1}, ..., S_{m+L-1} | m = 0, 1, 2, ...n\}. \tag{9}$$

Pre každý pár segmentov T_i a T_j môžeme vypočítať tzv. **correlation coefficent**, ktorý značíme ako $CC_{i,j}$. Ten nám určuje či sú jednotlivé segmenty spojené alebo nie. Z toho vyplýva, že correlation coefficient určuje výskyt hrán. Uzly v tejto sieti tvoria jednotlivé segmenty.

Pre každý pár segmentov T_i a T_j vypočítame correlation coefficient nasledovne:

$$CC_{ij} = \frac{\sum_{k=1}^{L} [T_i(k) - \langle T_i \rangle] \times [T_j(k) - \langle T_j \rangle]}{\sqrt{\sum_{k=1}^{L} [T_i(k) - \langle T_i \rangle]^2} \times \sqrt{\sum_{k=1}^{L} [T_j(k) - \langle T_j \rangle]^2}}, \tag{10}$$

kde $\langle T_i \rangle = \sum_{k=1}^{L} T_i(k)/L$. Jednotlivé elementy CC_{ij} sa nachádzajú v intervale $-1 \leq CC_{ij} \leq 1$.

Vzhľadom k tomu, že každý segment tvorí uzol, CC_{ij} popisuje pripojenie medzi dvojicami T_i a T_j. Zvolením kritickej hodnoty (z anglického critical value) r_c, môžeme jednotlivé hodnoty CC_{ij} konvertovať na hodnoty A_{ij}. Urobíme to nasledovným spôsobom :

$$A_{ij} = \begin{cases} 1, & (\,|C_{ij}| \geq r_c); \\ 0, & (\,|C_{ij}| < r_c). \end{cases} \tag{11}$$

Ak $A_{ij} = 1$, znamená to, že segmenty sú prepojené, ak $A_{ij} = 0$, medzi segmentmi nie je spojenie. Z toho je zrejmé, že výsledný graf bude neorientovaný. Dôležitú rolu hrá hodnota,

ktorú určíme pre r_c, pretože jej zmena priamo ovplyvňuje početnosť hrán v grafe. Čím je hodnota r_c menšia, tým viac hrán bude vo výslednom grafe a samozrejme naopak, čím bude táto hodnota vyššia, tým menší bude počet hrán. Nie je možné dopredu určiť, ktorá hodnota pre r_c bude vhodná. Na to aby sme zvolili túto hodnotu správne, musíme poznať dáta, ktoré máme k dispozícii. Ak by sme napríklad stanovili, že $r_c > 0.9$, mohla by nastať situácia, že graf nebude súvislý, čo je nežiadúci jav.

5.2.2 Recurrence networks

Posledným spôsobom transformácie časovej rady na sieť, ktorý si ukážeme v tejto práci je tzv. **recurrence network**. Anglické slovo recurrence v preklade znamená opakovanie, teda môžeme povedať, že táto metóda kladie dôraz na opakovania v rámci časovej rady. Poďme si recurrence networks popísať podrobnejšie podľa toho ako sa o nej dočítame v [20].

Na to aby sme získali recurrence network musíme získať takzvaný recurrence plot, čo je reprezentácia opakujúcich sa stavov systému v l-dimenzionálnom priestore. Je to testovanie všetkých párov vektorov $\vec{s_i}(i = 1, ...N, \vec{s} \in R^l)$, kde zisťujeme, či si navzájom sú alebo nie sú blízke. Recurrence plot vypočítame podľa (12).

$$R_{ij}(\epsilon) = \theta(\epsilon - d(\vec{s_i}, \vec{s_j})) \tag{12}$$

θ je v (12) takzvaná Heaviside function (táto funkcia rozhodne či budú výsledné uzly spojené alebo nie) a ϵ tvorí podmienku pre blízkosť vektorov. Blízkosť vektorov $d(\vec{s_i}, \vec{s_j})$ môže byť meraná rôznymi spôsobmi, najčastejšie cez Euklidovskú normu $d(\vec{s_i}, \vec{s_j}) = ||\vec{s_i} - \vec{s_j}||$. Z časovej rady $\{x(t_i)\}_{i=1...N}$ môžeme získať fázový priestor tak, že do vektoru vložíme takzvaný delay τ(oneskorenie v rámci časovej rady) tak, že položíme vektor $\vec{x_i} = (x_i, x_{i+\tau}, ..., x_{i+\tau(l-1)})$.

6 Implementácia vlastného nástroja

Hlavnou úlohou tejto bakalárskej práce je implementovať nástroj, ktorý by nám umožnil načítať časové rady, vytvárať z nich siete a následne tieto siete analyzovať. Taktiež je dôležité vytvoriť rozhranie, ktoré bude intuitívne ovládateľné a zamerané na čo najväčší user experience. Toto slovné spojenie v anglickom jazyku by sme mohli voľne preložiť ako užívateľský zážitok. Keďže tento preklad je dosť krkolomný, je lepšie skôr pochopiť myšlienku, ktorá sa za ním skrýva. V jednoduchosti ide o to, aký dojem zanecháva aplikácia v užívateľovi. Preto dôležitým prvkom tejto práce je aj vizualizácia, ktorá je jedným z najdôležitejších spôsobov budovania spomínaneho user experience. Konkrétne najmä vizualizácia sietí a vizualizácia dát získaných z týchto sietí.

Z toho vyplýva, že ako hlavné body si môžeme vytýčiť prácu s grafmi, sieťami a budovanie user experience. Riešením, ktoré tieto body umožňuje splniť je kombinácia frameworkov JUNG a JavaFX, o ktorých si povieme v nadchádzajúcich podkapitolách. Výhodou spomínanej kombinácie je to, že oba tieto frameworky sú vyvíjané v dobre známom a zdokumentovanom jazyku Java.

6.1 Java

Pojem Java je veľmi známy a rozšírený, avšak je potrebné si uvedomiť, že ním označujeme:

- programovací jazyk,
- platformu.

V tejto podkapitole si tieto dva termíny v jednoduchosti bližšie popíšeme.

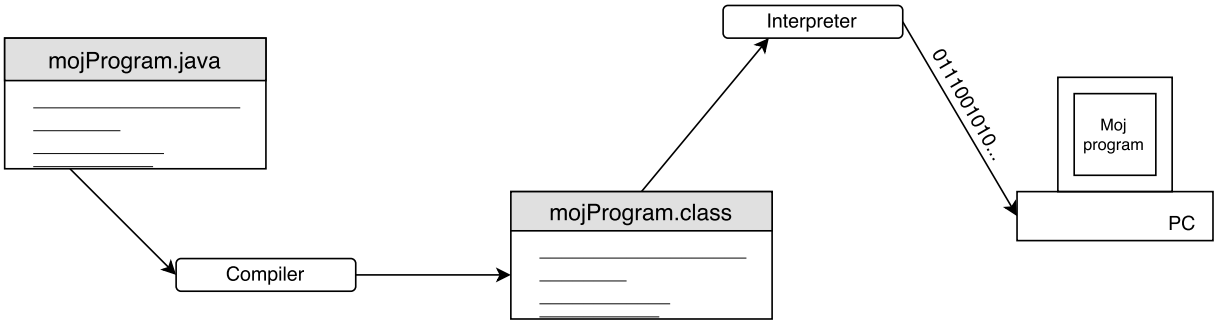
6.1.1 Java ako programovací jazyk

Java ako programovací jazyk podľa [9] je:

- jednoduchá (tým máme na mysli, že je zjednodušená práca programátora, napríklad odpadla práca s pointrami),
- objektovo orientovaná,
- robustná,
- prenosná,
- schopná využívať viac vláken (takzvaný multithreading- čo je dôležitá vlastnosť pri časovo náročných úlohách, ktoré chceme riešiť v tejto práci),
- dynamická,

- nezávislá na architektúre (ako môžeme vidieť aj na obrázku 5 compiler (prekladač) generuje súbor v takom formáte, že nezávisí na architektúre, vďaka čomu je možné takýto kompilovaný kód spustiť na rôznych systémoch, za predpokladu že obsahujú Java runtime).

Prekladač (v anglickom jazyku compiler) prekladá program do jazyka zvaného Java bytecode, čo je kód nezávislý na platforme. Tento kód je potom intepretovaný takzvaným interpreterom. Kompilácia sa prevedie iba raz, zatiaľ čo k interpretácii dochádza pri každom spustení programu. Tento proces môžeme názorne vidieť na obrázku 5.



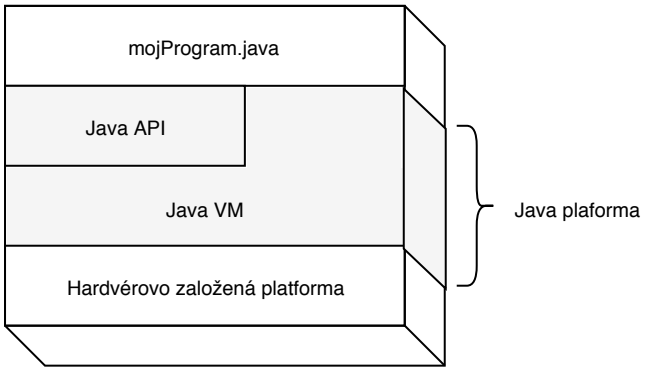
Obr. 5: Java - kompilovanie a interpretovanie

6.1.2 Java ako platforma

Platforma je hardvérové alebo softvérové prostredie, v ktorom beží program. Väčšinu platforiem je možné popísať ako kombináciu operačného systému a hardvéru. Java platforma sa odlišuje od väčšiny platforiem tým, že je len softvérovou platformou, ktorá beží na vrchu iných, hardvérovo-založených platforiem[9]. Java platforma pozostáva z dvoch komponentov:

- Java Virtual Machine (Java VM),
- Java Application Programming Interface (Java API).

Java API je rozsiahla kolekcia softvérových komponentov, ktoré poskytujú mnoho užitočných vlastností. Java API je rozdelené do knižníc spolusúvisiacich tried a rozhraní, tieto knižnice sú známe ako balíčky (anglicky packages). Na obrázku 6 môžeme vidieť, že Java API a Java VM oddeľujú program od hardvéru.



Obr. 6: Java API a Java VM oddeľujú program od hardvérových závislostí

6.2 JavaFX

Je zrejmé, že pre užívateľa hrá veľkú rolu už zmienená vizualizácia. Ako sa môžeme dočítať v [10] JavaFX je vhodná práve na takúto vizualizáciu. Pretože v [10] sa o JavaFX dozvieme, že ide o rodinu produktov, ktorú vyvinuli Sun Microsystems a jej primárnym účelom je jednoducho vytvárať grafické rozhranie pre užívateľa. Jadrom JavaFX je platforma Java, čo umožňuje, aby JavaFX pracovala s touto platformou bezchybne, vďaka čomu môže jednoducho ovplyvňovať existujúci Java kód.

6.3 JUNG

JUNG (skratka vychádza z anglického Java Universal Network/Graph Framework), ako sa dočítame v [25] je bezplatná, open-source knižnica, ktorá poskytuje bežný a rozšíriteľný jazyk na manipuláciu, analýzu a tiež vizualizáciu dát, ktoré môžu byť reprezentované ako graf, sieť. Tento framework je napísaný v programovacom jazyku Java, čo umožňuje aplikáciám využívajúcim JUNG využívanie rozsiahlych vstavaných možností, ktoré poskytuje Java Application Programming Interface (API). JUNG môže taktiež využívať existujúce Java knižnice tretích strán.

JUNG bol navrhnutý tak, aby podporoval rozličné reprezentácie entít a ich relácií. Za všetky spomeňme orientované a neorientované grafy. Aktuálna verzia JUNG taktiež zahŕňa implementácie viacerých algoritmov z teórie grafov, o ktorých sa môžeme dočítať aj v tejto práci.

V ukážke kódu 1 môžeme vidieť, ako jednoducho je možné vytvoriť graf, pridať doň uzly a následne ich spojiť hranou. JUNG využíva graf Graph<V,E>, pričom V označuje typ vrcholu a E typ hrany. V ukážke 1 si všimneme, že v nej vrchol tvorí objekt triedy Vertex a hranu objekt triedy Edge. Výhodou takejto implementácie je, že v objektoch môžeme uchovávať informácie, aké jednoduché dátové typy neumožňujú, čo je mimoriadne praktické. Vo všeobecnosti však V (vrchol) nemusí byť len objektom triedy, môže to byť aj jednoduchý dátový typ, obdobne to platí aj pre E (hranu). Všetko závisí len na tom, akú implementáciu požadujeme, aj preto je JUNG veľmi praktickým riešením.

```
// JUNG example

Graph<Vertex, Edge> g = new SparseMultigraph<Vertex, Edge>();
Vertex v1 = new Vertex(1);
Vertex v2 = new Vertex(2);

g.addVertex(v1);
g.addVertex(v2);

Edge e1 = new Edge(1);
g.addEdge(e1, v1,v2);
```

Výpis 1: Vytvorenie grafu pridanie uzlov a hrany medzi nimi

7 Nástroj pre analýzu časových rád pomocou sietí

V predchádzajúcich kapitolách sme postupne získali znalosti o grafoch, sieťach, o tom ako ich môžeme reprezentovať a tiež o ich vlastnostiach. Ďalej sme nabrali potrebné vedomosti o časových radách a spôsoboch ako ich prevádzať na siete. Tiež sme si ukázali knižnice, ktoré nám zjednodušia implementáciu všetkých týchto nadobudnutých znalostí. Teraz si teda môžeme ukázať, ako vyzerá činnosť nástroja postavená na týchto znalostiach a knižniciach.

7.1 Spustenie programu

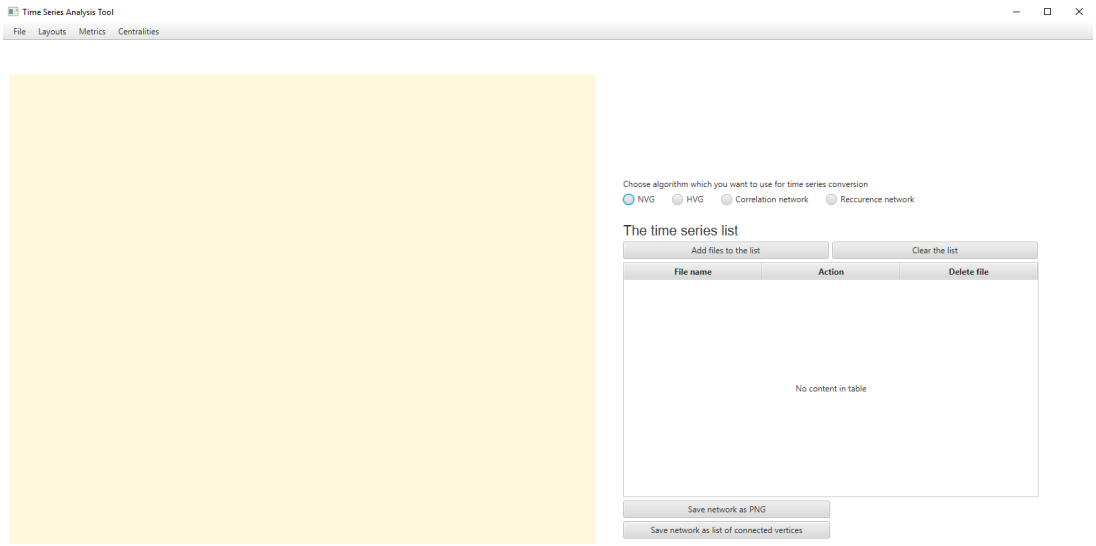
Funkčný program je uložený na CD priloženom k tejto bakalárskej práci pod názvom **TimeSeriesAnalysisTool.zip**. Po jeho rozbalení si môžeme všimnúť, že obsahuje zložku **lib**, kde sú uložené všetky externé knižnice, ktoré využíva JUNG. Program samotný spustíme tak, že dvakrát klikneme na súbor **TimeSeriesAnalysisTool.jar**, ktorý je hneď po spustení pripravený na používanie bez potreby inštalácie.

Minimálnou požiadavkou je nainštalovaná Java verzia 8. Na testovanie bol používaný notebook s procesorom Intel Core i5 3340M. Hoci program sám o sebe nie je veľmi náročný na výpočtový alebo grafický výkon, vyskytujú sa v ňom úlohy ako napr. výpočet priemeru siete či closeness centrality, ktorých výpočet na testovacích súboroch, ktoré sú priložené na CD, trval niekoľko desiatok sekúnd občas až niekoľko minút.

Program bol spúšťaný na systéme Windows 10 a počas jeho testovania sa nevyskytli žiadne ťažkosti, okrem problému s JavaFX, kde bol asi dvakrát zaznamenaný problém s prekreslením plátna, na ktoré sa vykresľujú grafy (inak povedané plátno sa nedalo prekresliť, takúto chybu jednoducho vyriešime ukončením a opätovným spustením programu).

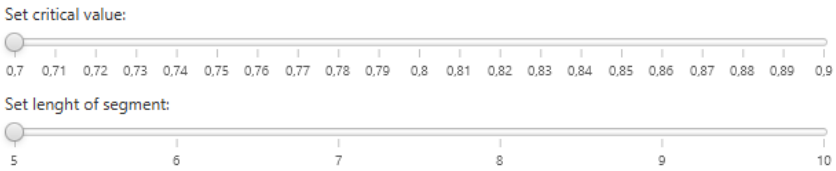
7.2 Základná obrazovka

Po spustení progamu vidíme aplikáciu, ktorej okno je rozdelené do niekoľkých častí (pozri obrázok 7). Hornú lištu tvorí hlavné menu so základnými možnosťami: **File, Layouts, Metrics, Centralities**. Na každú z týchto možností hlavného menu je možné kliknúť a zobraziť tak jej ďalšie možnosti.



Obr. 7: Základná obrazovka programu

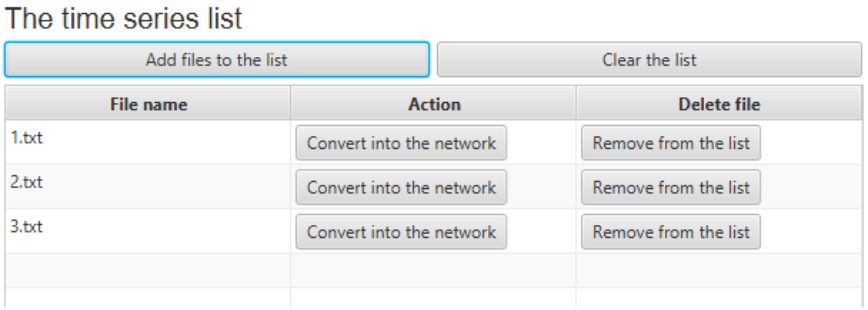
V pravo vidíme radio buttons, ktoré nám umožňujú zvoliť spôsob, akým sa budú časové rady transformovať na siete. Sú to spôsoby, ktoré sme v tejto práci rozoberali (NVG, HVG, correlation network a recurrence network). Pokiaľ nevyberieme žiadny z nich, využije sa NVG. Ak si vyberieme spôsoby correlation network alebo recurrence network, môžeme ešte nastaviť dĺžku segmentu a hodnotu, ktorá sa použije v podmienke pre získanie hrán (pozri obrázok 8). Pri Reccurence network môžeme navyše nastavovať ešte jeden parameter, ktorým je delay (pozri podkapitolu 5.2.2).



Obr. 8: Nastavenie parametrov pre tvorbu correlation network

Pod radio buttons a nápisom **The time series list** sa nachádza tabuľka do ktorej môžeme pridávať súbory s časovými radami, ktoré chceme prevádzať na sieť. Ak stlačíme tlačidlo **Add files to the list** otvorí sa dialógové okno, v ktorom môžeme prechádzať zložky a vybrať súbory s časovými radami. Program umožňuje len pridávanie súborov s príponou .txt, teda textové súbory. V prípade, že stlačíme tlačidlo **Clear the list** odstránia sa všetky zložky z tabuľky.

Po pridaní zložiek do tabuľky vidíme, že táto tabuľka má tri stĺpce. Prvý stĺpec obsahuje názov vybraného súboru, druhý tlačidlo **Convert into the network**. Ak toto tlačidlo stlačíme, rada na príslušnom riadku sa konvertuje na sieť. Tretí stĺpec obsahuje tlačidlo **Remove from the list**, ktoré umožňuje odstrániť z tabuľky súbor na príslušnom riadku (pozri obrázok 9).



Obr. 9: Tabuľka s časovými radami

Pod tabuľkou s časovými radami sa nachádzajú ďalšie dve tlačidlá. Prvé z nich, **Save network as PNG**, umožňuje uložiť obrázok vykreslenej siete (samozrejme, len ak nejaká sieť je vykreslená). Druhé tlačidlo, **Save network as list of connected vertices** umožňuje uloženie všetkých spojení medzi uzlami do textového súboru tak, že na každom riadku sú dva identifikátory navzájom spojených uzlov oddelené medzerou.

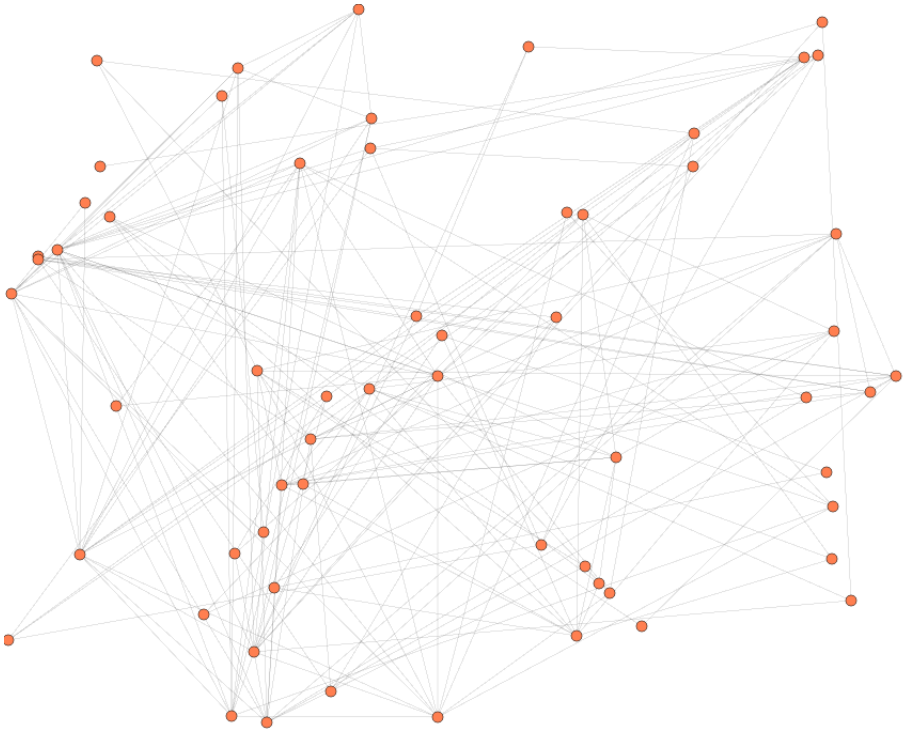
Najväčšiu časť okna aplikácie tvorí farebný štvoruholník, ktorý ohraničuje plochu, kde je možné vykresliť sieť získanú z časovej rady.

7.3 File

Už sme si spomenuli hlavné menu, ktoré tvorí horná lišta programu. Prvou z jeho položiek je File. Po kliknutí ma túto položku sa zobrazí možnosť **Exit Ctrl+X**. Ak klikneme na túto možnosť program je korektne ukončený. Všimnime si, že na ukončenie programu môžeme použiť aj stlačenie kombinácie kláves Ctrl+X.

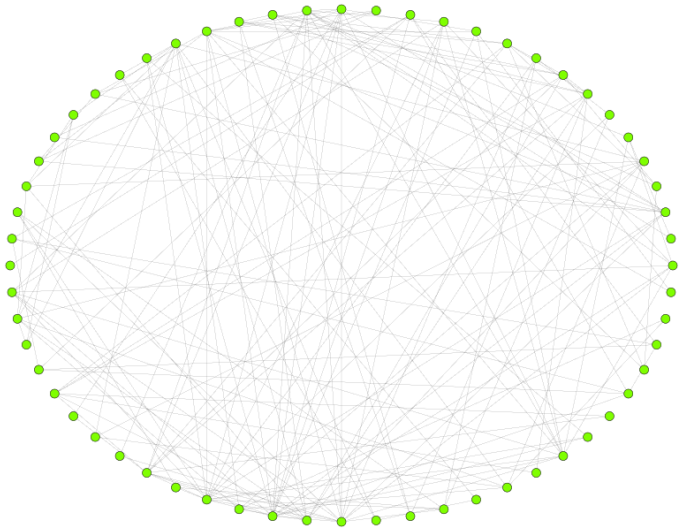
7.4 Layouts

Ako sme si povedali, vizualizácia je jedným z dôležitých faktorov pri budovaní tzv. user experience. Keďže tento nástroj umožňuje nie len transformáciu časovej rady na sieť, ale tiež vizualizáciu onej výslednej siete, prichádzajú na scénu tzv. **layouts** (toto anglické slovo by sme mohli preložiť ako rozloženie), ktoré rozhodujú o rozmiestnení uzlov. Prečo sú layouty dôležitou súčasťou vizualizácie si môžeme pozrieť na obrázku 10, kde nebol zvolený žiaden layout. Je zrejmé, že v takejto sieti sa orientujeme ťažko a nemá pre nás takmer žiadnu výpovednú hodnotu.



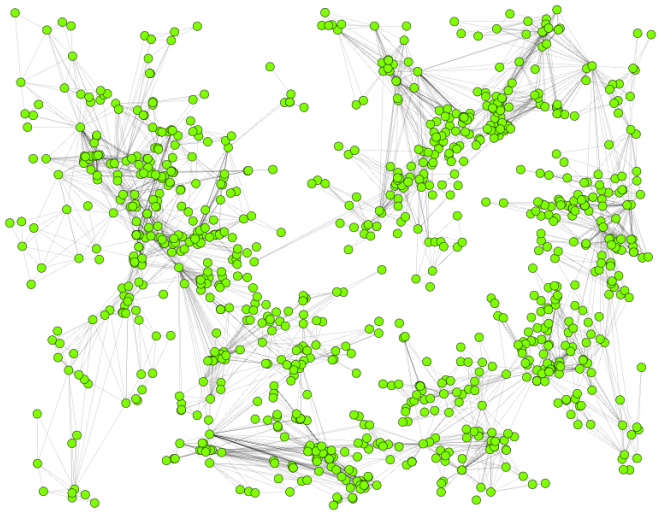
Obr. 10: Sieť bez použitia layoutu

Teraz, keď sme si ukázali význam layoutov, povedzme si v rýchlosti o tých, ktoré nájdeme v našom nástroji. Prvým je **CircleLayout**, ktorý rovnomerne rozmiestni uzly do tvaru kruhu (pozri obrázok 11).



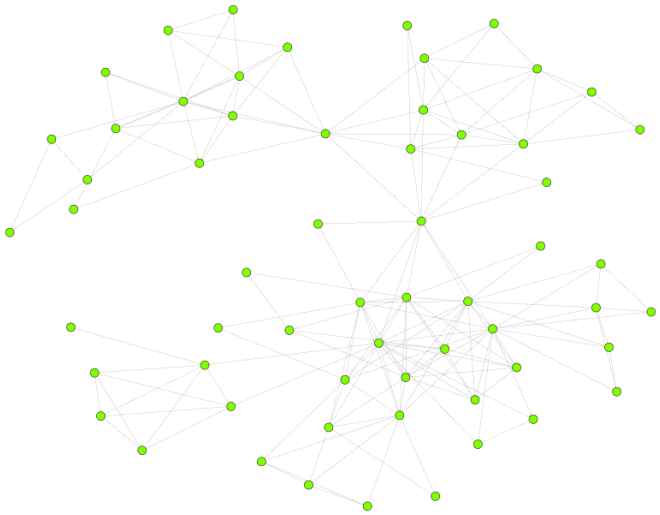
Obr. 11: Circular layout

Ďalším je **ISOMLayout**. Ten implementuje tzv. self-organizing map layout algorithm (teda voľne preložené algoritmus na seba-organizujúce rozmiestnenie), ktorý je založený na Meyerových metódach seba-organizujúcich grafov[21] (pozri obrázok 12).



Obr. 12: ISOM Layout

Taktiež je implementovaný aj **KKLayout**, ktorý implementuje tzv. Kamada-Kawai[15] algoritmus na rozloženie uzlov (pozri obrázok 13).



Obr. 13: KK Layout

Napriek tomu, že layouty sú veľmi užitočným nástrojom musíme si uvedomiť, že majú tiež svoje limity. Napríklad pri sieťach, kde sa vyskytujú radovo tisíce uzlov sa sieť môže stať napriek

ich využitiu neprehľadnou.

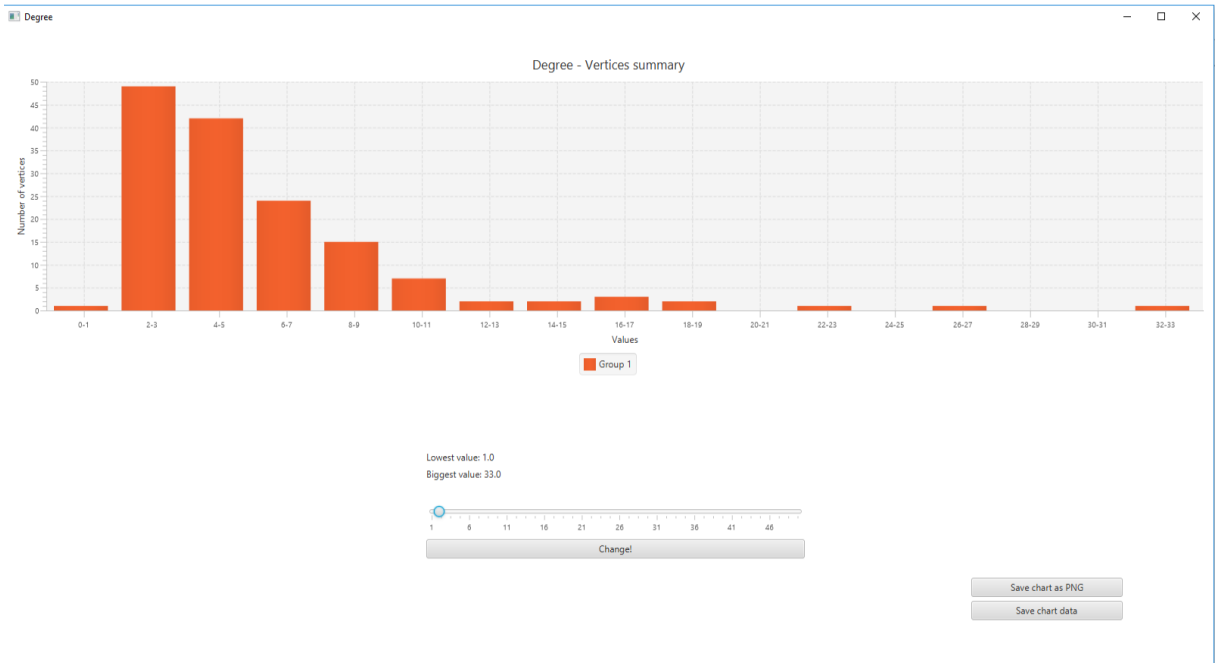
7.5 Metrics

Ďalšou položkou hlavného menu je položka Metrics. V nej nájdeme základné vlastnosti grafov, o ktorých sme si povedali v predošlých kapitolách. Konkrétne sa po kliknutí na položku menu Metrics zobrazia tieto možnosti:

1. **Average degree** vypočíta aký je priemerný stupeň uzla v grafe.
2. **Clustering coefficients** vypočíta zhlukovacie koeficienty a zobrazí pre ne graf.
3. **Diameter** vypočíta priemer siete.
4. **Degree** spočíta pre všetky uzly v grafe ich stupeň a hodnoty zobrazí v grafu takže môžeme odsledovať, koľko uzlov má príslušný stupeň.
5. **Degree distribution** vytvorí graf pre distribúciu stupňov.
6. **Number of vertices and edges** zobrazí údaj o počte uzlov a hrán. Tento údaj nie je tak celkom vlastnoťou grafu, ale pri veľkých neprehľadných sieťach môže byť takáto informácia užitočná.
7. **Network density** vypočíta hustotu siete.

Po kliknutí na niektorú s položiek clustering coefficients, degree a degree distribution, sa okrem zobrazenia hodnoty vytvorí aj graf. Ten má ešte ďalšie možnosti, tu si ukážme, čo sa stane, ak klikneme na položku Degree (pozri obrázok 14). Okrem toho, že v grafe vidíme, koľko uzlov má daný stupeň uzla, môžeme aj zmeniť rozsah osy-x v grafe. Na obrázku 14 vidíme, že rozsah je nastavený na 2. Teda jeden stĺpec reprezentuje 2 hodnoty. To znamená, že napríklad uzly so stupňom 0 a 1 sú v jednom stĺpci. Tento rozsah môžeme meniť pomocou slideru. Zmena nastane po kliknutí na tlačidlo **Change!**.

Na obrázku 14 tiež vidíme, že v pravo dole sa nachádzajú 2 tlačidlá. Prvé z nich, **Save chart as PNG**, uloží obrázok grafu na zvolenú destináciu. Druhé tlačidlo, **Save chart data**, uloží hodnoty do textového súboru pre jednotlivé uzly tak, že na každom riadku je hodnota pre jeden uzol. Hodnoty sú v súbore zoradené od najmenšej po najväčšiu, takže môžeme dodatočne rýchlo dohľadať, aká bola najvyššia či najnižšia hodnota.



Obr. 14: Graf početnosti výskytu daného stupňa uzlov

7.6 Centralities

Poslednou položkou hlavného menu je položka Centralities. Ak na ňu klikneme zobrazia sa nám nasledujúce možnosti:

- **Betweenness centrality**
- **Closeness centrality**
- **Degree centrality**
- **Eigenvector centrality**

Po kliknutí na ktorúkoľvek z týchto položiek sa zobrazí graf pre danú centralitu. Graf vyzerá pobdobne ako ten z obrázku 14. Rovnako poskytuje možnosti uloženia obrázka grafu, uloženia dát či prípadne zmeny rozsahu na osy-x (túto poslednú možnosť nemajú všetky vlastnosti).

8 Analýza časových rád

V predošlých kapitolách sme si získali teoretické základy pre tvorbu nástroja pre analýzu časových rád. Tento nástroj sme implementovali a popísali si jeho činnosť. Teraz prišiel čas tento nástroj využiť pri analýze dát. Tieto dáta sú v našom prípade, samozrejme, časové rady.

8.1 Dáta

Na priloženom CD je uložená zložka **TSAnalysis**. Po kliknutí na TSAnalysis sa nám zobrazia zložky **Data** a **AnalysisResults** Po rozkliknutí zložky Data sa nám zobrazia ďalšie 2 zložky **Best** a **Worst**. Zložka Best obsahuje textové súbory: **1.txt, 2.txt, 3.txt**. V každom z týchto súborov sú uložené časové rady tak, že na každom riadku je jedna hodnota časovej rady zaznamenaná v príslušnom čase. V každom jednom súbore je 4000 hodnôt. Podobne zložka Worst obsahuje 3 textové súbory. Konkrétne sú to súbory **1w.txt, 2w.txt, 3w.txt**. V týchto súboroch sú taktiež uložené hodnoty časových rád rovnakým spôsobom a v rovnakom množstve ako pri súboroch 1.txt, 2.txt a 3.txt.

V predošlej kapitole sme si ukázali ako načítať hodnoty zo súborov do nášho nástroja na analýzu časových rád a ako s nimi ďalej pracovať. Na analýzu časových rád v tejto BP využijeme súbory popísané v tejto podkapitole.

8.2 Zhodnotenie prevodov

Popísali sme si 4 prevody časovej rady na sieť. Každý z nich prevádza časovú radu na sieť na základe rôznych kritérii. V tabuľke 2 sa pozrime na to, koľko hrán bolo v grafe po danom prevode. Tabuľka obsahuje hodnoty pre súbory s dátami so zložiek Best a Worst. Môžeme pozorovať, že HVG je skutočne vždy podgrafom NVG pre danú časovú radu, tak ako sme sa to dozvedeli v podkapitole o prevode HVG. V tabuľke 2 vidíme, že prevodom HVG sme získali vždy o takmer 4 000 menej hrán, ako pri použití NVG.

Nepomerne najviac hrán v grafe vzniklo pri použití Correlation network (opäť pozri tabuľku 2), pretože sme museli nastavovať kritickú hodnotu r_c (pozri podkapitolu 5.2.1) tak, aby mal každý uzol stupeň aspoň jeden. Pretože práve nastavovaním tejto hodnoty zabezpečíme súvislý graf, zatiaľ čo pri NVG aj HVG je súvislosť grafu zabezbečená pre akúkoľvek časovú radu. Ďalším parametrom pri transformácii časovej rady na sieť spôsobom correlation network je dĺžka segmentu. O tom, aké hodnoty sme pre daný súbor použili pre nastavenie kritickej hodnoty a dĺžky segmentu, sa môžeme dočítať v každej zložke Correlation vytvorenej pre danú časovú radu, v súbore **ReadMe.txt**. Spravidla bola hodnota dĺžky segmentu nastavená na 10 a kritická hodnota buď 0,83 alebo 0,84.

Všimnime si, že v tabuľke chýba spôsob prevodu recurrence network, pretože ani pri nastavovaní rozdielnych dĺžok segmentov, či hodnôt pre delay alebo porovnávacej podmienky, sa nepodarilo získať súvislý graf. Pri zmenách podmienok dochádzalo k tomu, že pribúdalo veľké

množstvo hrán, no niektoré z uzlov zostávali stále odpojené. Tento spôsob teda nebol vhodný pre použitie na radách, ktoré sme analyzovali.

Tabuľka 2: Počet hrán získaných jednotlivými prevodmi zo súborov uložených v zložkách Best a Worst

Spôsob prevodu	1.txt	2.txt	3.txt	1w.txt	2w.txt	3w.txt
NVG	11 799	11 701	11 855	11 811	11 809	11 707
HVG	7 985	7 972	7 979	7 977	7 985	7985
Correlation network	18 505	18 786	18 733	18946	23 074	23 671

Taktiež pri výpočte vlastností sietí získanými rôznymi prevodmi sme získavali pomerne rôzne výsledky. Napríklad najväčšiu hustotu mali siete, ktoré sme získali prostredníctvom metódy correlation network, pretože ako sme si ukázali v tabuľke 2, takéto siete mali najväčší počet hrán. Pri pohľade na túto tabuľku je tiež zrejmé, že siete získané prevodom NVG mali druhú najväčšiu hustotu a najmenšiu hustotu mali siete získané prevodom HVG.

Priemer siete bol pri použití prevodu correlation network vždy rovný 7 alebo 8. Pre prevody NVG boli hodnoty priemeru siete vždy vyššie ako pre correlation network. Pre prevod HVG boli hodnoty priemeru siete najväčšie.

Rozdiely boli taktiež pri betweenness centralitách. V sieťach, ktoré boli vytvorené prostredníctvom correlation network malo len veľmi málo uzlov (jednotky) hodnotu betweenness centrality rovnú 0 a maximálne hodnoty nepresahovali 60 000. Zaujímavosťou bola sieť získaná zo súboru 3w.txt prevodom correlation network, kde ani jeden uzol v tejto sieti nemal betweenness centralitu rovnú 0. To v sieťach konvertovaných cez HVG a NVG malo spravidla viac než 1 000 uzlov hodnotu betweenness centrality rovnú 0 a nachádzalo sa v nich niekoľko uzlov s hodnotou betweenness centrality vyššou než 1 000 000 (maximum bolo takmer 10 000 000).

Odlišnosti medzi prevodmi boli aj v stupni uzlov. Pri correlation network a HVG boli hodnoty podobné a maximálny stupeň uzla v rámci týchto dvoch prevodov vo všetkých sieťach nikdy nebol väčší ako 26. Prevodom NVG vznikli grafy, kde boli oveľa väčšie najvyššie stupne uzlov (vždy aspoň 46, maximálne 78).

Pozrime sa ešte na rozdiely pri hodnotách zhlukovacieho koeficientu. Ak sme sa rozhodli pre prevod cez correlation network zhlukovací koeficient mal hodnotu 0 pre niekoľko desiatok uzlov a hodnotu 1 len pre najviac 6 uzlov (v rámci jednej siete). Ak sme robili prevod cez HVG alebo NVG hodnoty zhlukovacieho koeficientu vyzerali úplne odlišne. Vždy najviac 2 uzly mali hodnotu zhlukovacieho koeficientu rovnú 0 a vždy viac než 1 000 uzlov malo túto hodnotu rovnú 1.

8.3 Porovnanie jednotlivých časových rád vytvorených cez NVG

Ukázali sme si, že medzi prevodmi correlation network, NVG a HVG boli pomerne veľké rozdiely. Ak teda chceme porovnávať časové rady, je vhodné zamerať sa len na jeden spôsob prevodu

a porovnávať siete, ktoré ním vznikli. Poďme si teda porovnať siete, ktoré vznikli prevodom časových rád cez NVG.

V podkapitole 8.1 sme si popísali súbory, ktoré reprezentujú časové rady. Pre zložku Best to boli súbory 1.txt, 2.txt a 3.txt, pre zložku Worst zase 1w.txt, 2w.txt a 3w.txt. Pri porovnávaní jednotlivých časových rád z textu vypustíme príponu textového súboru .txt, teda ak povieme, že analyzujeme časovú radu 1, myslíme tým radu zo súboru 1.txt a obdobne to bude platiť aj pre zvyšné časové rady.

8.3.1 Porovnanie rád na základe hustoty siete

V tabuľke 3 vidíme hodnoty hustoty sietí, ktoré sme získali zo zadaných časových rád. Môžeme vidieť, že hustota siete bola pre všetky rady veľmi podobná.

Tabuľka 3: Hustota siete jednotlivých časových rád (NVG)

Časová rada	Hustota siete
1	0,001475
2	0,001463
3	0,001482
1w	0,001477
2w	0,001476
3w	0,001464

8.3.2 Porovnanie rád na základe priemeru siete

V tabuľke 4 vidíme hodnoty priemeru siete pre jednotlivé časové rady. Môžeme pozorovať, že siete získané z časových rád 2 a 3 majú oproti ostatným väčšiu hodnotu priemeru. Rovnaké hodnoty priemeru siete dosiahli rady 1 a 2w, kde bol priemer siete 11 a tiež rady 1w a 3w, kde tento priemer mal hodnotu 12.

Tabuľka 4: Priemer siete jednotlivých časových rád (NVG)

Časová rada	Priemer siete
1	11
2	16
3	15
1w	12
2w	11
3w	12

8.3.3 Porovnanie rád na základe stupňa uzla

Ak sa bližšie pozrieme na hodnoty stupňov uzlov jednotlivých časových rád nájdeme isté odlišnosti (pozri tabuľku 5). Niektoré z výsledných sietí mali aj tretiu najvyššiu hodnotu stupňa uzla, ktorý sa nachádzal v sieti, vyššiu ako najvyššia hodnota stupňa uzla v iných sieťach. Čo sa týka minimálnych stupňov uzla, 2 zo 6 sietí mali najnižšiu hodnotu stupňa uzla rovnú 2, vo zvyšných 4 bola najnižšia hodnota stupňa uzla 1, ale nedá sa hovoriť o nejakej odlišnosti, pretože ani v jednej zo sietí neboli viac než 2 uzly so stupňom 1 a vo všetkých 6 sieťach sa vyskytli stovky uzlov so stupňom 2.

Tabuľka 5: Minimálny a maximálny stupeň uzla v jednotlivých časových radách (NVG)

Časová rada	Najmenší stupeň uzla	3 navjyššie stupne uzla
1	1	72, 54, 50
2	2	47, 47, 45
3	2	76, 63, 54
1w	1	57, 56, 49
2w	1	78, 52, 49
3w	1	48, 46, 45

8.3.4 Porovnanie rád na základe zhlukovacieho koeficientu

Porovnanie časových rád na základe zhlukovacieho koeficientu môžeme vidieť v tabuľke 6, v druhom stĺpci vidíme najnižšiu hodnotu zhlukovacieho koeficientu uzla (zaokrúhlenú na 4 desatinné miesta), ktorý sa vyskytol v sieti, v treťom stĺpci nájdeme počet uzlov, ktoré mali hodnotu tohto koeficientu do 0,5 (vrátane) a v poslednom stĺpci koľko uzlov malo hodnotu zhlukovacieho koeficientu rovnú 1.

Časové rady 1w, 2w a 3w obsahovali vždy aspoň jeden uzol, ktorý mal hodnotu zhlukovacieho koeficientu rovnú 0. Rovnako je na tom aj časová rada 1. Časová rada 1 ale obsahuje až 1759 uzlov s hodnotou zhlukovacieho koeficientu rovnou 1 zatiaľ, čo rady 1w, 2w a 3w majú takýchto uzlov o 63, 15 a 62 menej.

Tabuľka 6: Hodnoty zhlukovacích koeficientov jednotlivých rád (NVG)

Časová rada	Minimum	Počet uzlov s hodnotou do 0,5	Počet uzlov s hodnotou 1
1	0	929	1759
2	0,0980	944	1759
3	0,0807	916	1715
1w	0 (2 uzly)	904	1696
2w	0	928	1744
3w	0	918	1697

8.3.5 Porovnanie rád na základe betweenness centrality

Ak sa pozrieme na betweenness centrality pre jednotlivé časové rady nájdeme isté odlišnosti (pozri tabuľku 7). V radách 1,2 a 3w bolo 11 až 12 uzlov, ktoré mali hodnotu betweenness centrality väčšiu ako 1 000 000. Ostatné 3 siete mali takýchto uzlov buď 8 alebo 9. Najvýznamenejší uzol z pohľadu betweenness centrality nájdeme v časovej rade 1w a má hodnotu 9 977 978. Ak si porovnáme toto maximum s maximálnymi hodnotami z ostatných časových rád najväčší rozdiel vidíme oproti rade 3w, kde najvýznamnešjí uzol z pohľadu betweenness centrality mal hodnotu 8 392 662, čo je značný rozdiel.

Tabuľka 7: Porovnanie betweenness centrality (NVG)

Časová rada	Počet uzlov s hodnotou 0	Počet uzlov s hodnotou >1 000 000	Maximum
1	1 761	12	9 382 108
2	1 760	11	8 958 734
3	1 716	8	9 796 482
1w	1 699	9	9 977 978
2w	1 746	9	9 818 352
3w	1 699	12	8 392 662

8.3.6 Porovnanie rád na základe closeness centrality

Poďme sa bližšie pozrieť ešte na closeness centralitu jednotlivých časových rád (pozri tabuľku 8, minimum a maximum v tejto tabuľke sú zaokrúhlené na 5 desatinných miest). Najväčšiu podobnosť v kritériách, ktoré sme si stanovili zaznamenali rady 3 a 1w, kde je počet uzlov s hodnotou closeness centrality menšou než 0,16 veľmi podobný, presnejšie 1823 a 1824. Rovnako sa tieto rady ponášajú aj v hodnote maximálnej centrality, kde rada 3 mala hodnotu 0,29710 a rada 1w 0,29783. To je naozaj veľká podobnosť, ak zvážime že najväčšiu maximálnu hodnotu closeness centrality nájdeme v rade 1 a má hodnotu 0,32430 a najmenšiu maximálnu hodnotu zas nájdeme v rade 2 a tá je len 0,25253. Ak sa pozrieme na minimálne hodnoty closeness centralít v rámci jednotlivých súborov vidíme, že rozdiely nie sú tak výrazné ako pri maximálnych hodnotách.

Pozornosť vzbudzuje aj rada 2, kde sa nachádzalo až 3 067 uzlov, ktorých closeness centralita nedosiahla hodnotu väčšiu alebo rovnú 0,16. To je výrazný rozdiel oproti všetkým zvyšným radám, obzvlášť ale voči rade 1, kde takýchto uzlov nájdeme len 1145.

Tabuľka 8: Porovnanie closeness centrality (NVG)

Časová rada	Minimum	Počet uzlov s hodnotou $< 0{,}16$	Maximum
1	0,12116	1145	0,32430
2	0,09274	3067	0,25253
3	0,09765	1823	0,29710
1w	0,11513	1824	0,29783
2w	0,11849	1260	0,31384
3w	0,11363	2189	0,28174

8.4 Porovnanie jednotlivých časových rád vytvorených cez HVG

V predchádzajúcej podkapitole sme si porovnávali vlastnosti sietí, ktoré boli vytvorené cez NVG. V tejto podkapitole sa zameriame na porovnávanie vybraných vlastností sietí, ktoré vznikli cez HVG.

Časové rady budeme skrátene značiť tak, ako sme to robili v predchádzajúcej podkapitole (teda 1, 2, 3 a 1w, 2w, 3w).

8.4.1 Porovnanie rád na základe hustoty siete

V tabuľke 9 vidíme hodnoty hustoty sietí, ktoré sme získali zo zadaných časových rád. Jednotlivé hodnoty sú veľmi podobné, dvakrát sa hustota siete rovnala 0,000997 a štyrikrát bola o sieť o niečo málo hustejšia, konkrétne mala hustota siete hodnotu 0,000998.

Tabuľka 9: Hustota siete jednotlivých časových rád (HVG)

Časová rada	Hustota siete
1	0,000998
2	0,000997
3	0,000998
1w	0,000997
2w	0,000998
3w	0,000998

8.4.2 Porovnanie rád na základe priemeru siete

V tabuľke 10 si môžeme prezrieť hodnoty priemeru siete pre jednotlivé časové rady. Všimnime si, že až 4 rady mali rovnaký priemer siete o hodnote 22, rada 1 mala priemer siete 19 a jednoznačne najvyšší priemer siete mala rada 2, a to až 28.

Tabuľka 10: Priemer siete jednotlivých časových rád (HVG)

Časová rada	Priemer siete
1	19
2	28
3	22
1w	22
2w	22
3w	22

8.4.3 Porovnanie rád na základe stupňa uzla

V tabuľke 11 si môžeme pozrieť ako dopadla analýza stupňa uzlov pre jednotlivé siete. Štyri rady mali vo výslednej sieti jeden uzol so stupňom uzla 1. Výnimkou je rada 1w, ktorá mala takéto uzly 2 a rada 3 nemala ani jeden uzol, ktorý by bol stupňa 1. Pozornosť vzbudzuje rada 2, ktorá mala aj svoj najvyšší stupeň uzla menší alebo rovný akémukoľvek inému z troch najvyšších stupňov uzlov spomedzi ostatných rád.

Tabuľka 11: Minimálny a maximálny stupeň uzla v jednotlivých časových radách (HVG)

Časová rada	Najmenší stupeň uzla	3 navjyššie stupne uzla
1	1	21, 20, 20
2	1	18, 17, 17
3	2	23, 19, 19
1w	1	19, 19, 19
2w	1	24, 20, 18
3w	1	23, 18, 18

8.4.4 Porovnanie rád na základe zhlukovacieho koeficientu

V tabuľke 12 vidíme analýzu zhlukovacích koeficientov pre jednotlivé rady. V treťom stĺpci nájdeme počet uzlov do 0,5 s tým, že sú tu zarátané aj uzly ktoré mali zhlukovací koeficient rovný práve 0,5.

V každej rade sa nachádzal aspoň jeden uzol, ktorý mal hodnotu zhlukovacieho koeficientu rovnú 0. Keď si sčítame počty uzlov v treťom a štvrtom stĺpci je zrejmé, že väčšina uzlov v každej jednej sieti má hodnotu zhlukovacieho koeficientu buď 0,5 a menšiu, alebo má túto hodnotu rovnú 1.

Tabuľka 12: Hodnoty zhlukovacích koeficientov jednotlivých rád (HVG)

Časová rada	Minimum	Počet uzlov s hodnotou do 0,5	Počet uzlov s hodnotou 1
1	0	1 777	1 343
2	0 (2 uzly)	1 780	1 346
3	0	1 765	1 325
1w	0 (2 uzly)	1 798	1 309
2w	0	1 763	1 339
3w	0	1 800	1 333

8.4.5 Porovnanie rád na základe betweenness centrality

Pozrieme sa teraz na betweenness centrality pre jednotlivé časové rady (pozri tabuľku 13). Iba v rade 1 sme nenašli aspoň 21 uzlov, ktoré mali hodnotu betweenness centrality väčšiu než 1 000 000. Najviac takýchto uzlov mala rada 2, a to celkom 25, no mala druhú najnižšiu maximálnu hodnotu betweennes centrality (8 739 039). Až štyri rady obsahovali uzol, ktorého hodnota betweennes centrality prekročila hranicu 9 200 000.

Tabuľka 13: Porovnanie betweenness centrality (HVG)

Časová rada	Počet uzlov s hodnotou 0	Počet uzlov s hodnotou >1 000 000	Maximum
1	1 344	19	9 460 916
2	1 347	25	8 739 039
3	1 325	22	9 733 631
1w	1 311	21	9 258 599
2w	1 340	22	9 207 339
3w	1 334	22	8 610 392

8.4.6 Porovnanie rád na základe closeness centrality

Poďme sa ešte zamerať na closeness centralitu jednotlivých časových rád (pozri tabuľku 14, minimum a maximum v tejto tabuľke sú zaokrúhlené na 5 desatinných miest).

Na prvý pohľad je zrejmé, že rada 2 má jednoznačne najviac uzlov s hodnotou closeness centrality menšou než 0,1, je ich 3 041. Zvyšné rady sa k tomuto číslu nepribližujú, rada 1 mala takýchto uzlov dokonca len 1093. Práve rada 1 má aj uzol s maximálnou hodnotou closeness centrality, na druhej strane, rada 2 má túto maximálnu hodnotu najnižšiu. Rozdiel medzi maximálnou hodnotou closeness centrality rady 1 a 2 je 0,04632. Rady 1 a 2 musíme zmieniť aj pri minimálnej hodnote closeness centrality, rada 1 má túto hodnotu najvyššiu a rada 2 najnižšiu. Rozdiel je však o niečo menší ako tomu bolo pri maximálnych hodnotách, tentokrát je hodnota rozdielu 0,02367.

Tabuľka 14: Porovnanie closeness centrality (HVG)

Časová rada	Minimum	Počet uzlov s hodnotou < 0,10	Maximum
1	0,07439	1 093	0,20125
2	0,05063	3 041	0,15493
3	0,06230	2 322	0,17106
1w	0,06639	1 575	0,18179
2w	0,06352	1 321	0,18921
3w	0,06685	2 568	0,16126

8.5 Porovnanie jednotlivých časových rád vytvorených cez correlation network

Prezreli sme si porovnania časových rád, ktoré vznikli cez NVG a HVG. V tejto podkapitole si porovnáme časové rady, ktoré boli konvertované spôsobom correlation network. Ako sme si povedali pri correlation network určujeme parametre, ktorými sú dĺžka segmentu a kritická hodnota. V podkapitole 8.2 sme si povedali, ako sme tieto parametre nastavili. Je potrebné si uvedomiť, že siete, ktoré vznikali prevodom correlation network mali o niečo menší počet uzlov ako siete, ktoré vznikali cez NVG a HVG. Konkrétne to bolo **3 991** uzlov zatiaľ čo NVG a HVG mali pre rovnaké časové rady uzlov presne 4 000. Prečo tomu tak je môžeme overiť v podkapitolách o NVG (5.1.1), HVG (5.1.2) a correlation network (5.2.1).

Aj pri hodnotení rád transformovaných na sieť pomocou correlation network budeme tieto rady označovať rovnako ako v analýze cez NVG a HVG, teda 1, 2, 3, 1w, 2w a 3w.

8.5.1 Porovnanie rád na základe hustoty siete

V tabuľke 15 si môžeme pozrieť hodnoty hustoty jednotlivých sietí, ktoré sme získali zo zadaných časových rád. Istú odlišnosť je badať na radách 2w a 3w, ktorých výsledné siete sú o niečo hustejšie (majú viac hrán).

Tabuľka 15: Hustota siete jednotlivých časových rád (correlation)

Časová rada	Hustota siete
1	0,002324
2	0,002359
3	0,002353
1w	0,002380
2w	0,002898
3w	0,002973

8.5.2 Porovnanie rád na základe priemeru siete

V tabuľke 16 si môžeme prezrieť hodnoty priemeru siete pre jednotlivé časové rady. Vidíme, že priemery siete majú v 4 prípadoch zhodnú hodnotu 8. Iba rady 2w a 3w majú hodnotu priemeru siete rovnú 7.

Tabuľka 16: Priemer siete jednotlivých časových rád (correlation)

Časová rada	Priemer siete
1	8
2	8
3	8
1w	8
2w	7
3w	7

8.5.3 Porovnanie rád na základe stupňa uzla

V tabuľke 17 môžeme pozorovať hodnoty stupňov uzlov pre jednotlivé rady. Rady obsahovali len málo uzlov, ktorých stupeň bol 1. Najviac uzlov stupňa 1 sme objavili v sieti vzniknutej z časovej rady 2, bolo ich tam 7. Rada 3w nemala ani jeden uzol, ktorý by mal stupeň 1. Najnižším stupňom v tejto rade bol stupeň uzla 2 a tento malo 5 uzlov. Rada 3w mala zároveň 3 najvyššie stupne uzla rovné 25, čo je viac než v hociktorej inej rade v rámci troch najvyšších stupňov uzlov. Jedinú výnimku tvoril jeden uzol v rade 2w, ktorý mal stupeň až 26 a teda bol najvyšší spomedzi všetkých stupňov uzlov v rámci všetkých rád.

Tabuľka 17: Minimálny a maximálny stupeň uzla v jednotlivých časových radách (correlation)

Časová rada	Najmenší stupeň uzla	3 navjyššie stupne uzla
1	1 (2 uzly)	21, 20, 19
2	1 (7 uzlov)	23, 22, 21
3	1 (5 uzlov)	24, 22, 21
1w	1 (3 uzly)	23, 23, 21
2w	1 (1 uzol)	26, 24, 23
3w	2 (5 uzlov)	25, 25, 25

8.5.4 Porovnanie rád na základe zhlukovacieho koeficientu

V tabuľke 18 môžeme pozorovať hodnoty zhlukovacích koeficientov. V druhom stĺpci je zaznamenaný počet uzlov, ktoré mali hodnotu zhlukovacieho koeficientu rovnú 0, v treťom stĺpci počet uzlov, ktoré mali túto hodnotu menšiu ako 0,17 a v poslednom stĺpci nájdeme počet uzlov, ktoré mali hodnotu zhlukovacieho koeficientu rovnú 1.

Môžeme vidieť, že najviac sa líšia časové rady 2w a 3w. Tieto rady majú pomerne jednoznačne najmenej uzlov s hodnotou zhlukovacieho koeficientu rovnou 0, ďalej majú najmenej uzlov, kde je hodnota tohto koeficientu menšia ako 0,17 a tiež majú najmenej uzlov, pre ktoré by mal koeficient hodnotu 1. Rada 3w dokonca nemá ani jediný uzol s hodnotou zhlukovacieho koeficientu 1.

Tabuľka 18: Počty uzlov so zhlukovacím koeficientom podľa rôznych kritérií (correlation)

Časová rada	hodnota = 0	hodnota < 0,17	hodnota = 1
1	96	1 349	3
2	82	1 366	3
3	88	1 286	4
1w	83	1 336	6
2w	20	1 150	1
3w	27	1 130	0

8.5.5 Porovnanie rád na základe betweenness centrality

Poďme si teraz porovnať rady na základe betweenness centralít výsledných sietí (pozri tabuľku 19).

Pozornosť pútajú rady 2w a 3w, ktoré majú najmenej uzlov s betweenness centralitou rovnou 0. 2w ich má 2 a 3w dokonca 0. Tieto rady napriek tomu majú jednoznačne najväčšie množstvo uzlov, ktorých hodnota betweenness centrality nepresiahla 15 000 a taktiež majú suverénne najnižšiu maximálnu hodnotu betweenness centrality.

Zvyšné rady sú pomerne podobné, avšak rada 3 je zaujímavá tým, že má jednoznačne najvyššiu hodnotu betweeness centrality.

Tabuľka 19: Porovnanie betweenness centrality (correlation)

Časová rada	Počet uzlov s hodnotou 0	Počet uzlov s hodnotou < 15 000	Maximum
1	5	2 185	54 593
2	10	2 162	54 498
3	9	2 192	59 663
1w	9	2 186	53 099
2w	2	2 533	42 054
3w	0	2 523	42 268

8.5.6 Porovnanie rád na základe closeness centrality

Ako poslednú vlastnosť si ešte prezrieme closeness centralitu a pozrieme sa na to, ako sa jej hodnoty líšili naprieč jednotlivými radami (pozri tabuľku 20).

Opäť si musíme na prvý pohľad všimnúť rady 2w a 3w, tieto rady majú len 6 respektíve 5 uzlov, ktoré by mali hodnotu closeness centrality menšiu ako 0,2. Majú najvyššie hodnoty ako medzi minimálnymi tak medzi maximálnymi hodnotami closeness centralít.

Tabuľka 20: Porovnanie closeness centrality (correlation)

Časová rada	Minimum	Počet uzlov s hodnotou $< 0{,}20$	Maximum
1	0,16034	420	0,22878
2	0,16184	350	0,22767
3	0,16195	396	0,22983
1w	0,17067	343	0,23076
2w	0,18738	6	0,24473
3w	0,19113	5	0,24780

8.6 Výsledky

Všetkých 6 súborov s uloženými časovými radami, o ktorých sme sa dočítali v podkapitole 8.1 môžeme prevádzať na sieť a následne pre takúto sieť vypočítavať vlastnosti. Všetky vlastnosti (pre všetky siete), ktoré nástroj umožňuje získať, sú uložené na priloženom CD. Dostaneme sa k nim tak, že postupne rozklikávame zložky **TSAnalysis** a **AnalysisResults**. V zložke AnalysisResults nájdeme rovnako ako v zložke Data zložky **Best** a **Worst**. Obe obsahujú rovnako štrukturované zložky a súbory. Ak rozklikneme napr. Best vidíme tri zložky s názvami **1, 2 a 3** (ak rozklineme zložku Worst, vidíme zložky **1w, 2w, 3w**). Ak otovríme ktorúkoľvek z nich vidíme ďalšie 3 zložky, tentokrát s názvami **Correlation, HVG, NVG**. Napríklad v zložke na ceste TSAnalysis\AnalysisResults\Best\1\NVG nájdeme vypočítané vlasnosti získané zo súboru TSAnalysis\Data\Best\1.txt, ktoré boli transformované na sieť spôsobom NVG. Teda číslo zložky korešponduje s číslom dát v súbore s dátami.

9 Záver

Táto bakalárska práca mala vytýčených niekoľko základných na seba nadväzujúcich cieľov. Prvým bolo naštudovať a popísať problematiku sietí. Tento cieľ bol splnený. V úvodných kapitolách sme si popísali a osvojili základné termíny z teórie grafov a tiež vlastnosti, ktoré tieto grafy majú a ako je možné ich vypočítať. Ďalej sme si tiež ukázali ako je možné grafy reprezentovať.

Druhým cieľom bolo naštudovať a popísať možnosti prevodu časových rád na sieť. Tento cieľ bol rovnako splnený, z dostupnej literatúry sme si predstavili celkom 4 rôzne prevody, hoci NVG a HVG majú mnoho spoločného.

Tretím cieľom bolo naprogramovať a popísať nástroj, ktorý umožní načítanie časovej rady, jej prevod na sieť a základnú analýzu takto vzniknutej siete. Aj tento cieľ bol splnený. Najskôr sme si ukázali nástroje, ktoré nám s takouto prácou môžu pomôcť a tie sme využili pri implementácii nášho vlastného nástroja. Tento nástroj umožňuje načítanie časových rád, prevádzať ich na siete a tieto siete ďalej analyzovať. Nástroj taktiež umožňuje ukladanie takýchto výsledkov do súborov.

Posledným cieľom bolo porovnať dodané časové rady. Analýzu časových rád sme si popísali v poslednej kapitole tejto bakalárskej práce, kde sme si prezreli vypočítané vlastnosti pre zadané časové rady. Všetky údaje, ktoré sme získali zo zadaných časových rád, sú, ako sme si už povedali, uložené na priloženom CD.

Celkovo teda môžeme hodnotiť túto prácu ako úspešnú, keďže sa podarilo naplniť všetky vytýčené ciele. Samozrejme, človek nedosiahne pokrok tým, že sa uspokojí so splnením svojich cieľov. Na to, aby dosiahol progres, si musí klásť neustále nové a vyššie ciele. Inak tomu nie je ani v tejto bakalárskej práci. Bolo by určite možné a zaujímavé získať vedomosti o nejakom ďalšom prevode časovej rady na sieť, tento následne implementovať do nástroja, a tak získať nové siete, ktoré by bolo možné ďalej analyzovať a porovnávať.

Literatúra

[1] Mathieu Bastian, Sebastien Heymann, Mathieu Jacomy, et al. Gephi: an open source software for exploring and manipulating networks. *Icwsm*, 8:361–362, 2009.

[2] Vladimir Batagelj and Andrej Mrvar. Pajek-program for large network analysis. *Connections*, 21(2):47–57, 1998.

[3] Stefano Boccaletti, Vito Latora, Yamir Moreno, Martin Chavez, and D-U Hwang. Complex networks: Structure and dynamics. *Physics reports*, 424(4):175–308, 2006.

[4] Stephen P Borgatti, Martin G Everett, and Linton C Freeman. Ucinet. In *Encyclopedia of social network analysis and mining*, pages 2261–2267. Springer, 2014.

[5] Danah Boyd. Why youth (heart) social network sites: The role of networked publics in teenage social life. *MacArthur foundation series on digital learning–Youth, identity, and digital media volume*, pages 119–142, 2007.

[6] Danah Boyd. Social network sites as networked publics: Affordances, dynamics, and implications. In *A networked self*, pages 47–66. Routledge, 2010.

[7] Danah M Boyd and Nicole B Ellison. Social network sites: Definition, history, and scholarship. *Journal of computer-mediated communication*, 13(1):210–230, 2007.

[8] Peter J Brockwell and Richard A Davis. *Time series: theory and methods*. Springer Science & Business Media, 2013.

[9] Mary Campione, Kathy Walrath, and Alison Huml. *The Java tutorial: a short course on the basics*, volume 1. Addison-Wesley Professional, 2001.

[10] Jim Clarke, Jim Connors, and Eric J Bruno. *JavaFX: developing rich Internet applications*. Pearson Education, 2009.

[11] Nicole B Ellison, Charles Steinfield, and Cliff Lampe. The benefits of facebook "friends:" social capital and college students' use of online social network sites. *Journal of computer-mediated communication*, 12(4):1143–1168, 2007.

[12] Mark S Handcock, David R Hunter, Carter T Butts, Steven M Goodreau, and Martina Morris. statnet: Software tools for the representation, visualization, analysis and simulation of network data. *Journal of statistical software*, 24(1):1548, 2008.

[13] Robert A Hanneman and Mark Riddle. Introduction to social network methods, 2005.

[14] Derek Hansen, Ben Shneiderman, and Marc A Smith. *Analyzing social media networks with NodeXL: Insights from a connected world*. Morgan Kaufmann, 2010.

[15] Tomihisa Kamada, Satoru Kawai, et al. An algorithm for drawing general undirected graphs. *Information processing letters*, 31(1):7–15, 1989.

[16] Petr Kovář. Úvod do teórie grafů. *Vysoká škola báňská. Technická univerzita. Ostrava*, 2012.

[17] Lucas Lacasa, Bartolo Luque, Fernando Ballesteros, Jordi Luque, and Juan Carlos Nuno. From time series to complex networks: The visibility graph. *Proceedings of the National Academy of Sciences*, 105(13):4972–4975, 2008.

[18] Vito Latora and Massimo Marchiori. Vulnerability and protection of infrastructure networks. *Physical Review E*, 71(1):015103, 2005.

[19] Alexandra Marin and Barry Wellman. Social network analysis: An introduction. *The SAGE handbook of social network analysis*, 11, 2011.

[20] Norbert Marwan, Jonathan F Donges, Yong Zou, Reik V Donner, and Jürgen Kurths. Complex network approach for recurrence analysis of time series. *Physics Letters A*, 373(46):4246–4254, 2009.

[21] Bernd Meyer. Self-organizing graphs—a neural network perspective of graph layout. In *International Symposium on Graph Drawing*, pages 246–262. Springer, 1998.

[22] Daniel Miller. Social networking sites. *Digital anthropology*, pages 146–61, 2012.

[23] Mark Newman. *Networks: an introduction*. Oxford university press, 2010.

[24] Angel M Nuñez, Lucas Lacasa, Jose Patricio Gomez, and Bartolo Luque. Visibility algorithms: A short review. In *New Frontiers in Graph Theory*. InTech, 2012.

[25] Joshua O'Madadhain, Danyel Fisher, Padhraic Smyth, Scott White, and Yan-Biao Boey. Analysis and visualization of network data using jung. *Journal of Statistical Software*, 10(2):1–35, 2005.

[26] Michael Small, Jie Zhang, and Xiaoke Xu. Transforming time series into complex networks. In *International Conference on Complex Sciences*, pages 2078–2089. Springer, 2009.

[27] Duncan J Watts and Steven H Strogatz. Collective dynamics of 'small-world'networks. *nature*, 393(6684):440–442, 1998.

[28] Yue Yang and Huijie Yang. Complex network-based time series analysis. *Physica A: Statistical Mechanics and its Applications*, 387(5-6):1381–1386, 2008.

[29] Jie Zhang and Michael Small. Complex network from pseudoperiodic time series: Topology versus dynamics. *Physical review letters*, 96(23):238701, 2006.